PASSED

Group 4, Lab 3

Kevin Neville, Gustav Sternelöv, Vuong Tran

19 februari 2016

Assignment 1: Cluster sampling

An opinion pool is assumed to be performed in several locations of Sweden by sending interviewers to this location. Of course, it is unreasonable from the financial point of view to visit each city. Instead, a decision was done to use random sampling without replacement with the probabilities proportional to the number of inhabitants of the city to select 20 cities. Explore the file population.xls. Note that names in bold are counties, not cities.

1.1. Import necessary information to R.

1.2. Use uniform random number generator to create a function that selects 1 city from the whole list by the probability scheme offered above (do not use standard sampling functions present in R).

1.3. Use the function you have created in step 2 as follows:

a. Apply it to the list of all cities and select one city

b. Remove this city from the list

c. Apply this function again to the updated list of the cities

d. Remove this city from the list

e. .. and so on until you get exactly 20 cities.

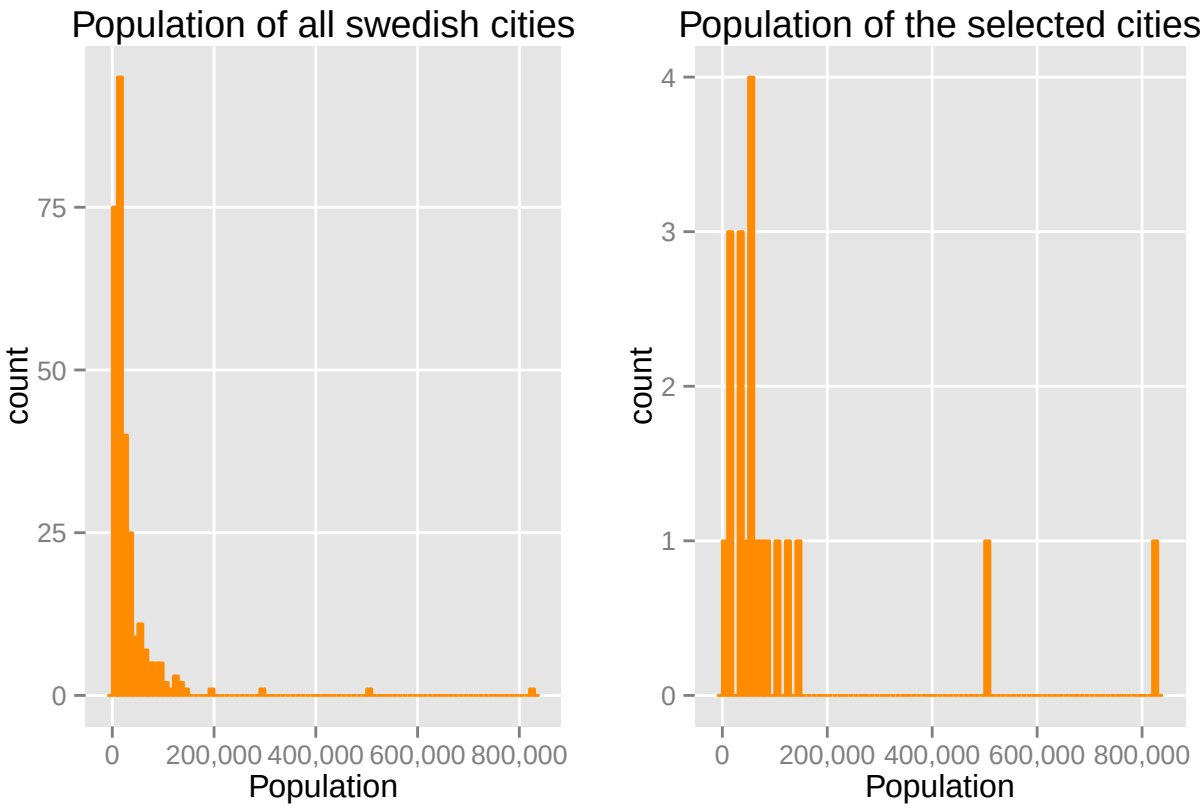
1.4. Run the program. Which cities were selected? What can you say about the size of the selected cities?

```
##         Cities Population Above_median
## 1  Katrineholm      32303         TRUE
## 2   Södertälje      85270         TRUE
## 3    Stockholm     829417         TRUE
## 4     Göteborg     507330         TRUE
## 5     Nyköping      51209         TRUE
## 6    Falköping      31419         TRUE
## 7        Luleå      73950         TRUE
## 8         Lund     109147         TRUE
## 9    Linköping     144690         TRUE
## 10     Svedala      19625         TRUE
## 11   Jönköping     126331         TRUE
## 12   Norrtälje      55927         TRUE
```

```
## 13    Forshaga      11401        FALSE
## 14     Ödeshög       5314        FALSE
## 15     Kungälv      40727         TRUE
## 16   Uddevalla      51518         TRUE
## 17        Håbo      19452         TRUE
## 18       Falun      55685         TRUE
## 19        Täby      63014         TRUE
## 20   Sandviken      36978         TRUE
```

From the output above it is clearly that the selected cities are the ones with larger populations. The majority of the selected has a larger population than the median of Swedish cities populations.
This is the expected result since the probability of getting picked is correlated with the size of the population for every city. As a example of this we see that the biggest city, Stockholm, is selected which is the city with the highest probability of getting picked.

1.5. Plot one histogram showing the size of all cities of the country. Plot another histogram showing the size of the 20 selected cities. Conclusions?



From the two histograms we can see that the two cities with the highest number of population is drawn in our sample. We can also notice that the histogram of the selected cities is more skewed towards higher populations. This is expeted since their probability is higher of being selected in our sample.

Assignment 2: Different distributions

The double exponential (Laplace) distribution is given by formula:

$$DE(\mu, \alpha) = \frac{\alpha}{2}\, exp(-\alpha\, |x - \mu|)$$

2.1. Write a code generating double exponential distribution DE(0,1) from U(0,1) by using inverse CDF method. Explain how you obtained that code step by step. Generate 10000 random numbers from this distribution, plot the histogram and comment whether the result looks reasonable.

The double exponential (Laplace) distribution is given by the formula:

$$DE(\mu, \alpha) = \frac{\alpha}{2} * \exp(-\alpha|x - \mu|)$$

The formula can be rewritten as, with $\mu = 0\ and\ \alpha = 1$:

$$DE(0,1) = \frac{1}{2} * \exp(-|x|)$$

$$f(x) = \begin{cases} \frac{1}{2} * e^{-x}, & x \geq 0\ (1) \\ \frac{1}{2} * e^{x}, & x < 0\ (2) \end{cases}$$

$$(1): 1 - \frac{1}{2}\int_{x}^{\infty} e^{-t}\, dt = 1 - \frac{1}{2}\,[-e^{-t}]_{x}^{\infty} = 1 - \frac{e^{-x}}{2}$$

$$(2): \frac{1}{2}\int_{-\infty}^{x} e^{t}\, dt = \frac{1}{2}\,[-e^{-t}]_{-\infty}^{x} = \frac{e^{x}}{2}$$

$$F(x) = \begin{cases} 1 - \frac{e^{-x}}{2}, & x \geq 0\ (1) \\ \frac{e^{x}}{2}, & x < 0\ (2) \end{cases}$$

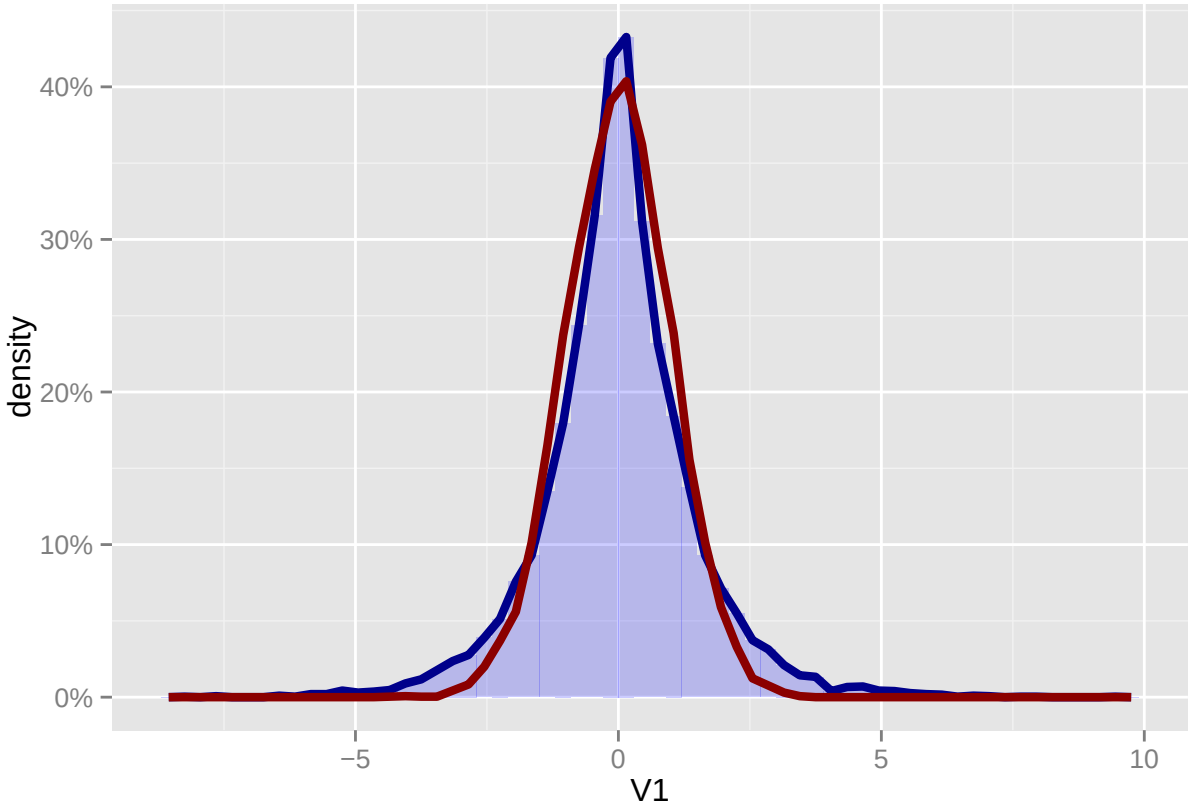
To find F_x^{-1}, one need to solve x for both the equations.

$$(1): y = 1 - \frac{e^{-x}}{2} \rightarrow e^{-x} = 2(1-y) \rightarrow x = -\ln(2(1-y))$$

$$(2): y = \frac{e^{x}}{2} \rightarrow e^{x} = 2y \rightarrow x = \ln(2y)$$

Thus the transform from U to X becomes:

$$X = \begin{cases} -\ln(2(1-U)), & U \geq 0.5 \\ \ln(2U), & U < 0.5 \end{cases}$$



The double exponential distribution is expected to have fatter tails than the normal distribution since the density is expressed in terms of the absolute difference from the mean, rather than the squared difference from the mean.

In the histogram is the double exponential values generated from the inverse CDF method visualized with the blue area and the blue line. For comparsion with the normal distribution is also an sample of 10000 N(0,1) values visualized with the red line. It can easily be seen that the double exponential distribution has a higher peak and has fatter tails, hence the results appears to be reasonable.

2.2. Use Acceptance/rejection method with DE(0,1) as majorizing density to generate N(0,1) variables. Explain step by step how this was done. How did you choose constant c in this method? Generate 2000 random numbers N(0,1) using your code and plot the histogram. Compute the average rejection rate R in the acceptance/rejection procedure. What is the expected rejection rate ER and how close is it to R? Generate 2000 numbers from N(0,1) using standard rnorm procedure, plot the histogram and compare the obtained two histograms.

The following steps are performed in order to use the acceptance/rejection method with DE(0,1) as majorizing density to generate N(0,1) variables.
¤ First, the value of c is calculated.
This is done by using the Laplace distributed values from *2.1*. The value of c must be higher than all ratios of $f_y(x)$ and $f_x(x)$ where f_y is the pdf for the double exponential distribution and f_x is the pdf for the normal distribution.

$$c > \frac{f_x(x)}{f_y(x)}$$

Then, the value of c is determined to be 1.315489.
¤ The first step of the acceptance/rejection is to generate Y.
This is done with the inverse CDF method explained in *2.1*
¤ Then the generated Y values is inserted into $f_y(Y)$ and $f_x(Y)$.
This is done by using the probability density function for f_y, the double exponential distribution, and f_x, the normal distribution.
¤ Generate a U[0,1] value.
¤ Test if Y should be accepted or rejected.
If the Y should be kept or not is decided with the following rule. If

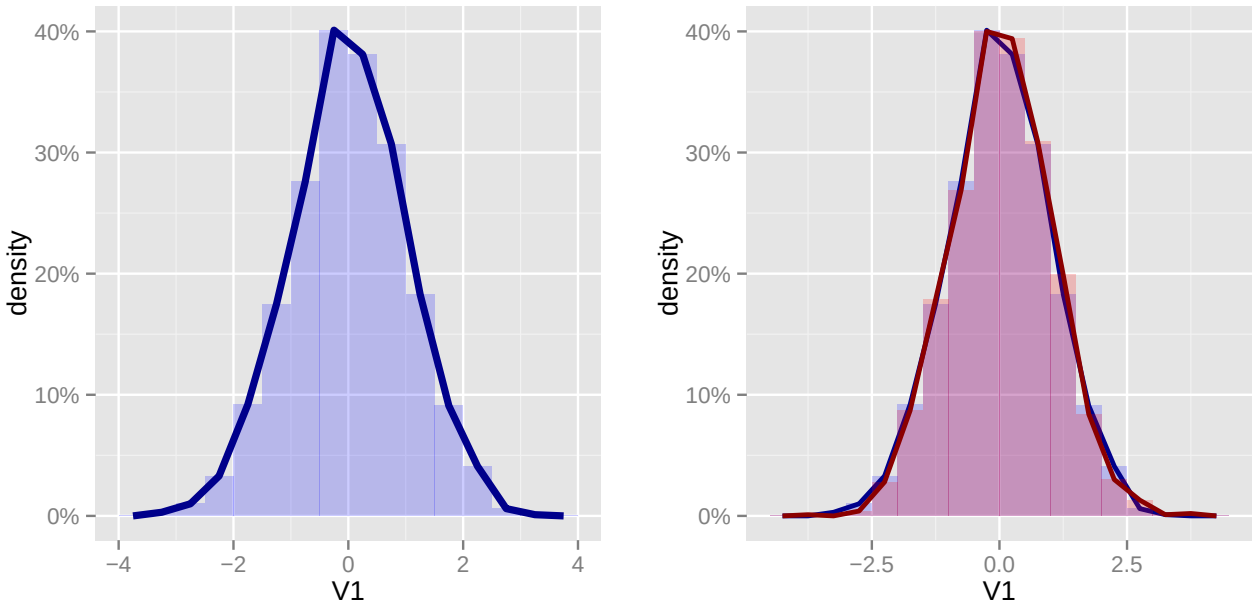
$$U \leq \frac{f_x(Y)}{C * f_y(Y)}$$

keep Y, else return to step one and generate a new Y value.

The implemented acception/rejection algorithm is tested and 2000 N(0,1) values are generated. In the histogram below are these values illustrated with the blue line and the red line is 2000 N(0,1) values generated with *rnorm* function.
The acceptance rate is $\frac{1}{c}$, hence the rejection rate is $1 - \frac{1}{c}$. The expected rejection rate then is

$$1 - \frac{1}{1.315489} = 0.24$$

The 2000 N(0,1) values generated with the acceptance/rejection method required 2604 Y values to be generated. So 604 values were rejected, resulting in an rejection rate of 0.23. In other words are the obtained rejection rate very close to the expected rejection rate. Hence, the chosen value for c seem to be reasonable. In the histogram below to the left are the values generated from the acception/rejection method plotted. In the histogram to the right are these values, the blue line, compared to 2000 N(0,1) values, the red line, generated with the *rnorm* function.



As can be seen in the histogram to the right are the obtained histograms very similar to each other. Both the acceptance/rejection algorithm and the *rnorm* function produces values that seem to be N(0,1) distributed.

Contributions:

Most of the codes in assignment one comes from Kevins file, but the plots is taken from Gustavs code. The text is compiled by everyone which we did togehter. In assignment two we used Vuongs derivatives of the equations. The text is complied togehter by the group. Most of the code is taken from Gustav.

Appendix - R-code

```
library("gdata")
library("ggplot2")

# Importing data
mydf <- read.xls("/Users/Kevin/Desktop/Computational-statistics/Lab3/population.xls", encoding="latin1"

# Cleaning data
mydf <- mydf[-c(1:8),c(1:3)]
colnames(mydf) <- c("Cities_code", "Cities", "Population")

# Removing bold_cities
index <- NA
pop_no_comma <- NA
for(i in 1:nrow(mydf)) {
index[i] <- length(unlist(strsplit(as.vector(mydf$Cities_code[i]), "")))
pop_no_comma[i] <- as.numeric(paste(unlist(strsplit(as.vector(mydf$Population[i]), ",")), collapse=""))
}

bold_cities <- which(index==2)
mydf$Population <- pop_no_comma
population <- mydf[-bold_cities, ]
CityFunc <- function(k, population){

  exclude <- NA
  temp_population <- population

  for(i in 1:k){
    set.seed(i)
    val <- runif(1,0,1)
      temp_population$upper_limits <- cumsum(temp_population$Population / sum(temp_population$Populatio
      temp_population$lower_limit <- c(0, temp_population$upper_limits[-length(temp_population$upper_li
      select_this_city <- which(temp_population$lower_limit < val & val < temp_population$upper_limit)
      exclude[i] <- which(population[,2] == as.vector(temp_population[select_this_city, 2])) # Save the
      temp_population <- temp_population[-select_this_city, ] # Remove the selected city from temp_data
  }
  res <- data.frame(cbind(population[exclude, 2:3], Above_median=population[exclude, 3] > median(popula
  return(res)
}
CityFunc(20, population = population)
options(scipen = 999)
library(gridExtra)
library(scales)
All <- ggplot(population, aes(Population)) + geom_histogram(binwidth=10000, fill="darkorange", col="dar
  scale_x_continuous(labels=comma) + theme(panel.grid.minor=element_blank())

Sel <- ggplot(CityFunc(20, population = population), aes(Population)) + geom_histogram(binwidth=10000,
  scale_x_continuous(labels=comma) + theme(panel.grid.minor=element_blank()) + labs(x = "Population") +

grid.arrange(All, Sel, ncol=2)
## Assignment 2 ##
# 10 000 random uniform values
```

```r
set.seed(190216)
randoms <- runif(10000, 0, 1)
randIndex <- randoms < 0.5

# The uniformed values are transformed by using the inverse CDF method.
# The transformed values follows a DE(0,1) distribution.
DEval1 <- data.frame(V1=log(2*randoms[randIndex]))
DEVal2 <- data.frame(V1= -log(2-2*randoms[!randIndex]))
DEval <- data.frame(rbind(DEval1, DEVal2))
set.seed(160219)
NormTest <- data.frame(V1=rnorm(10000, 0, 1))


ggplot(DEval, aes(V1,..density..)) + geom_histogram(fill="blue", alpha=0.2,binwidth=0.3) +
  scale_y_continuous(labels = percent_format()) +
  geom_freqpoly(size=1.5, col="darkblue",binwidth=0.3) +
  geom_freqpoly(data = NormTest,size=1.5, col="darkred",binwidth=0.3)
c <- 1.315489
i <- 0
j <- 1
y <- data.frame(V1=0)
set.seed(311015)
while (length(y[,1])<2000) {
  i <- i+1
  #set.seed(i)
  Utest <- runif(1,0,1)
  if(Utest < 0.5){
    DEy <- log(2*Utest)
  }else{
    DEy <- -log(2-2*Utest)
  }
  fy_y <- 1/2 * exp(-1 * abs(DEy-0))
  fx_y <- dnorm(DEy, 0, 1)
  #set.seed(i+10)
  Uselect <- runif(1,0,1)
  if (Uselect <= fx_y/(c*fy_y)) {
    y[j,] <- DEy
    j <- j+1
  }else{
    j <- j
  }
}
set.seed(311015)
NormTest <- data.frame(V1=rnorm(2000, 0, 1))
ARnm <- ggplot(y, aes(V1,..density..)) + geom_histogram(fill="blue", alpha=0.2,binwidth=0.5) +
  scale_y_continuous(labels = percent_format()) +
  geom_freqpoly(size=1.5, col="darkblue",binwidth=0.5)

RNnm <- ggplot(y, aes(V1,..density..)) + geom_histogram(fill="blue", alpha=0.2,binwidth=0.5) +
  scale_y_continuous(labels = percent_format()) +
  geom_freqpoly(size=1, col="darkblue",binwidth=0.5) +
  geom_histogram(data = NormTest, fill = "red", alpha = 0.2,binwidth=0.5)+
  geom_freqpoly(data = NormTest,size=1, col="darkred",binwidth=0.5)
```

```r
grid.arrange(ARnm,RNnm, ncol=2)
```

```
## NA
```